SYSTEM AND METHOD FOR DETECTING AND VERIFYING DIGITIZED CONTENT OVER A COMPUTER NETWORK

BACKGROUND OF THE INVENTION

This invention relates to the field of online digital content distribution and more particularly, to a system and method for facilitating music distribution and authentication over a communications network.

The internet has created a highway for users and companies to share digitized content. Online services allow digitized content stored on servers to be shared by multiple users via the internet. Online services also allow users to play digitized content stored in an Internet-connected repository.

It is advantageous for online service providers to detect and verify whether or not the user has a physical copy of digitized content, such as a CD or DVD, prior to allowing the user access to the digitized content.

SUMMARY OF THE INVENTION

The present invention system identifies and authenticates digitized content, such as compact audio disc (hereinafter "CD-Audio," or "CD") residing in a CD-Audio-compatible drive of a computer and verifies that the CD is authentic or an exact replica. However, the present invention is not limited to CD verification. In certain embodiments of the invention, digitized content stored on DVDs or other medium including a physical disc, disc drive, or in solid state memory devices, may be verified. The invention may be practiced in a number of electronic devices, including personal computers, disc players such as CD players and DVD players, and other electronic devices. In certain embodiments according to the present invention, a verification database is created from a set of master CDs. The verification database contains records of CDs and a

corresponding table-of-contents, also known as a table -of-contents identifier, (hereinafter "TOC") and corresponding selected audio data from the CD.

After the verification database is created, verification of a CD to the master CD may be performed. The CD is first identified by matching the TOC from the CD against the verification database. Using the TOC data the system identifies one or more master CDs with a similar TOC. The identified CDs are then authenticated by matching selected audio data from the CD against the verification database created from a set of master CDs.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1. details the overall architecture of the system;

FIG. 2. details the verification database creation system;

FIG. 3. depicts key client and server operations during identification and verification.

DETAILED DESCRIPTION OF THE INVENTION

In the following embodiments of the invention, common reference numerals are used to represent the same components. If the features of an embodiment are incorporated into a single system, these components can be shared and perform all the functions of the described embodiments.

In FIG. 1, a Server 111 and a Client 121 communicate with each other via a communications network 113 for the purpose of identifying and authenticating or verifying digital content. In one embodiment of the present invention, a user inserts a CD for verification in a CD Reader 123, the Client 121 controls the CD Reader 123 as necessary to acquire data from the CD. The Client 121 communicates the data to the Server 111.

The Client 121 is a general purpose personal computer programmed to read CDs from the CD reader 123. The Client 121 is typically located at a remote location 117 which is connected to the network 113 via a communications link 119. In one embodiment the Client 121 is used by an Internet user computing from their home or office. The communications link 119 may be a dial-in modem connecting to an internet service provider or a broad-band service such as DSL or cable internet access.

The Server 111 is programmed to receive information from the Client 121 for verification with information stored in the Verification Database 106. The Server 111 is typically programmed to facilitate multiple connections from Clients 121 and 129, each with a CD Reader 123 and 131 respectively, and connected to the Network 113 via a communications link 115. The Clients 121 and 129 are also connected to the Network 113 via communications likes 119 and 127 respectively. Typically the Server 111 and the Verification Database 106 are located at a Server Facility 101 to optimize system performance. In another embodiment, the Server 111 may be located in a separate facility from the Verification Database 106. In a preferred embodiment of the invention the Server 111 is a high performance micro-computer running the UNIX operating system.

Before the Server 111 can identify and verify CDs for the Client 121, the corresponding CD data must be stored in the Verification Database 106. An Encoding Computer 103 is programmed to read master CDs from a CD reader 105 and store data about the CD in the Verification Database 106. Alternatively, data about the CD is computed from digital audio files stored on a computer that contain a copy of the audio data found on a master CD.

The Verification Database 106 is comprised of a Verification Table 107 and an Identification Table 109. Creation of the Verification Database 106 is accomplished by computing and storing entries in the database for each CD to be identified and verified

by the Encoding Computer 103. Each database entry comprises several elements of identification and verification data which are computed from the TOC and audio data extracted from an original, authentic CD title.

In one embodiment of the present invention the various components and computers of the system communicate with each other using a general connection-oriented protocol such as the Transmission Control Protocol / Internet Protocol (TCP/IP), which is described in Internetworking with TCP/IP, 3d. ed., Douglas E. Comer, (1995), which is hereby incorporated by reference. However, the present invention is not limited to TCP/IP or any other particular network architecture, software or hardware which may be described herein. The principles of the invention apply to other communications protocols, network architectures, hardware and software which may come to compete with or even supplant the state of the art at the time of the invention.

In FIG. 2 the Verification Database 106 is comprised of two tables: an Identification Table 109 and a Verification Table 107. Each entry in the Identification Table 109 comprises a subset of the TOC data from the corresponding CD title, and multiple subsets of TOC data are stored for each corresponding CD title. This data is used during the identification phase of the disc verification procedure to quickly locate CDs that have a TOC similar to the CD being identified. The Identification Table 109 is comprised of the following fields:

Disc Identifier - A value assigned during database creation that uniquely identifies the CD.

TOC Identifier - A hash value computed from the CD TOC.

Disc Length - Total length (in blocks) of the audio portion of the CD.

First Track Length - Length (in blocks) of the first audio track on the CD.

Last Track Length - Length (in blocks) of the last audio track on the CD.

Shortest Track Length - Length (in blocks) of the shortest audio track on the CD.

Longest Track Length - Length (in blocks) of the longest audio track on the CD.

Disc Songprint - An identifying value computed from the CD audio data.

Once created, the entire Identification Table 109 may be sorted by and stored in ascending or descending order using the value of the Disc Length field to facilitate faster look ups.

In FIG. 2 the Verification Table 107 is comprised of identification and verification data that is both copied and computed from the corresponding CD title by the Encoding Computer 103. This data is used during the disc verification procedure to test the identity and validity of the CD being verified. The Verification Table 107 is comprised of a number of individual keys. Each key is computed by the Encoding Computer 103 and stored in the Verification Table 107. The value of each key is derived from audio data read from a certain region of the CD by the CD Reader 105, as instructed by the Verification Table 107. An entry in the Verification Table 107 is comprised of the following fields:

Descriptive Data - Includes CD title and artist.

Disc Identifier - A value assigned during database creation that uniquely identifies the CD.

TOC Identifier - A hash value computed from the CD TOC.

Disc Songprint - An identifying value computed from the CD audio data.

Track Data - The following fields are included for each track:

Length - Length (in blocks) of the track

Alignment Guide Data - Data derived from the audio data of the track

Title - Textual title of the track

Track Songprint - An identifying value computed from the audio data of the track.

Key Data - The following fields are included for each key:

Track - The number of the track which includes the key region.

Offset - The location of the key region within the specified track.

Alignment Guide Data - Data derived from the audio data in the key region.

Hash Data - A hash value computed from the audio data in the key region.

Key Songprint - An identifying value computed from the audio data in the key region.

The Encoding Computer 103 calculates a TOC identifier. A TOC identifier is computed from the CD TOC data by computing a cryptographic hash value using SHA-1 (Secure Hash Algorithm) of the concatenation of the lengths, in blocks, of each track on the CD represented as 4-byte values and truncating the resulting 20-byte hash value to 8 bytes.

The Encoding Computer 103 calculates a songprint. A songprint is a 128-byte value that represents the spectral content of a region of a digital audio recording. It is computed by the following steps:

The two stereo channels are averaged to produce a single channel.

The songprint region is divided into 512-byte chunks. Any partial chunks are discarded. Additionally, for each chunk, the following computations are made:

The data is detrended by computing a linear regression and removing the result.

A Hanning window is applied to the data.

A Fast Fourier Transform (FFT) is computed for the data.

The DC component of the result is discarded.

The squared magnitudes of each of the remaining spectral components are computed.

The spectral components are divided into groups of 4 and averaged to produce 64 spectral components.

Each of the first 64 bytes of the songprint value is computed as follows:

The mean of each of the 64 spectral components resulting from each chunk is computed.

The mean is converted to a logarithmic value by computing the log10 and multiplying by 10. Values less than $1x10^{-20}$ are assigned the value –200.

The resulting dB value is scaled and shifted then converted to an unsigned integer byte value. The scale and shift amounts are chosen to maximize resolution within the range (0-255) expressible in a single byte.

Each of the final 64 bytes of the songprint value is computed as follows:

The standard deviation of each of the 64 spectral components resulting from each chunk is computed.

The standard deviation is converted to a logarithmic value by computing the log10 and multiplying by 10.

The resulting dB value is scaled and shifted then converted to an unsigned integer byte value. The scale and shift amounts are chosen to maximize resolution within the range (0-255) expressible in a single byte.

The Encoding Computer 103 uses the region to generate the songprint; the region varies between the Disc and Track Songprints and the Key Songprints. The Encoding Computer 103 selects the songprint region by first identifying the length of

any "silent" audio at the beginning of the track. This is accomplished by reading 4096-byte blocks of audio data and computing a root-mean-square (RMS) of the amplitude of the samples (the two channels are averaged for each sample during this computation).

The end of the initial silent portion of a track is located by finding the first block that has an RMS amplitude which exceeds the predefined threshold. The beginning of the songprint region is then computed by adding a predefined offset. The length of the songprint region is a predefined value.

For Track Songprints, the RMS amplitude threshold for detecting the end of the initial silence is 0.001. The predefined offset from the end of the initial silence to the beginning of the songprint region is 30 seconds (30*75*2352 bytes). The predefined length of the songprint region is 5 seconds (5*75*2352 bytes).

A Disc Songprint is defined as the Track Songprint for the first track on the CD. The Key Songprint region is the same as the key region. This is because no silence detection or region offset is applied. The Key Songprint region length, like the key region length, is 4096 bytes.

The Encoding Computer 103 generates a Track Alignment Guide. A Track Alignment Guide comprises a 4-byte sample search value and a 4-byte hash value computed from the audio data block midway through the track. If the track is an odd number of blocks in length, the block at the midpoint is used. If the track is an even number of blocks in length, the block immediately after the midpoint is used.

The 4-byte sample search value is the first 4 bytes of the audio data block. The 4-byte hash value is computed by hashing the first 64 bytes of the audio data block using the SHA-1 algorithm and truncating the result to 4 bytes.

The Encoding Computer 103 generates a Key Alignment Guide. A Key Alignment Guide comprises eight 2-byte samples taken from the audio data contained

within a key region. The samples are taken at 292-sample intervals starting with the first sample contained within the key region (samples offsets 0, 292, 584, 876, 1168, 1460, 1752, and 2044).

The Encoding Computer 103 generates a Key Hash Data. Key Hash Data is computed by hashing all the bytes contained within the key region using the SHA-1 algorithm and storing the entire 20-byte hash result.

In FIG.1., the verification procedure is accomplished through a sequence of processes and messages that are exchanged between a Client 121 in which the CD to be verified is located, and a Server 111 which queries a Verification Database 106 as shown in more detail in FIG. 3. The Client 121 and the Server 111 communicate using a network 113. In another embodiment of the invention, the Server 111 may contain the Verification Database 106 internally.

In FIG. 3 block 301 the client begins the verification process. Typically the client may be programmed to begin the process whenever a disc is inserted into the CD reader 123. In block 303, the client reads the Table-of-Contents data from the CD using the appropriate features of the client operating system. Also in block 303, the TOC data is formatted and placed into the Initial Request message. The Initial Request message may be formatted to contain subsets of the TOC data, or the complete TOC data. Also in block 303, the client computes the Disc Songprint for the CD according to the algorithm specified earlier and places it into the Initial Request message, which is sent to the server.

In block 305 Initial Request Processing is performed by the server upon receipt of an Initial Request message from the client. The server receives the Initial Request message from the client and proceeds to extract the TOC and Disc Songprint. The server, using the Identification Table, then locates the entry that best matches the TOC and Disc Songprint provided by the client. The server performs a binary search of the

Identification Table (which is sorted by Disc Length) to find the entry that most nearly matches the disc length specified in the TOC.

In block 305, beginning with the entry in the Identification Table identified above, the server compares all neighboring entries to the TOC and Disc Songprint provided by the client. For each entry, the server first tests whether the disc length specified by the TOC and the disc length recorded in the table entry are within a specified limit. The server then computes the root-mean-square (RMS) of the differences between each of the first-, last-, shortest-, and longest-track fields of the table entry and the corresponding data from the TOC. The RMS difference must fall within a specified limit. Finally, the server computes the RMS difference between the corresponding data points (each of the 128 bytes) in the table entry songprint and the Disc Songprint provided by the client.

In block 307, the server selects the entry in the Identification Table that has the smallest RMS difference between the songprint and the one provided by the client, the Best Match. If that RMS difference does not fall within a specified limit, the verification fails and the server constructs a Disc Not Found message in block 309. If the RMS for the Best Match falls within the specified limit, the process proceeds to block 311.

In another embodiment, the server computes the RMS difference between the client-provided and database-provided values for each of the disc length, the first-, last-, shortest-, and longest-track fields, and each of the 128 bytes of the songprint and weights those individual differences to compute a single weighted-difference value representing the overall fit between the client-provided and database-provided data. The server selects as the Best Match the entry in the Identification Table that results in the smallest weighted-difference. In an alternate embodiment, the server selects all the entries which have weighted-difference values less than a predefined threshold and attempts to verify each of these Matches.

In block 311, the server locates the entries in the Verification Table corresponding to the Best Match values. Because each entry in the Verification Table contains a large number of usable verification keys, in block 313, the server selects a smaller subset of key candidates that will be used in the current disc verification. The subset is selected using a pseudo-random sequence that is seeded with the client network address and the current time reduced to half-day resolution (i.e., the same key candidates will be selected for a given network address during a given half day).

In block 313, the key region (the region of audio data on the CD from which each key was computed) is enlarged using the pseudo-random sequence so that the actual key region starts at a pseudo-random offset within the enlarged key region. In addition to the real key candidates selected from the Verification Table entry, a set of decoy keys are also generated, also using the address/time-seeded pseudo-random sequence (i.e., the same decoy key candidates will be generated for a given network address during a given half day). The decoy keys are chosen so as not to overlap the audio data regions from which the real keys are derived. In an alternate embodiment, a random sequence is used to select and adjust keys and generate decoy keys so that each verification attempt by a client causes the server to specify a different set of verification regions.

The server then proceeds to construction of a Verification Response message. The Verification Response message is constructed by the server in response to an Initial Request message from the client. It is also constructed in response to a Verification Request message from the client that fails the verification test as discussed below.

Also in block 313, from the key candidates selected during Initial Request Processing, the server selects one or more keys and includes the offset and length data for each key region in the Verification Response message. A key candidate is used

only once during a single disc verification. When all key candidates have been used and the disc has not been successfully verified, the verification fails.

From the decoy key candidates selected during Initial Request Processing, the server selects one or more decoy keys and includes the offset and length data for each key region in the Verification Response message. A decoy key candidate is used only once during a single disc verification. The server generates enough decoy keys during Initial Request Processing so that the decoy keys are not exhausted before the disc keys.

The state of the disc verification process is encrypted and included in the Verification Response message. This includes the presumed identity of the disc, the selected key candidates, the generated decoy key candidates, and the key usage information (which keys/decoys have been requested from the client). The state information is returned to the server by the client in the Verification Request message and is decrypted by the server and used to restore the state of the verification process. The Track Alignment Guide data stored in the Verification Database entry is included in the Verification Response message. Finally, the Verification Response message is sent to the client.

In block 315, for each of the key regions requested by the server, the client determines in which track the region resides, checks the track alignment, and reads the requested data. The client begins track alignment by reading a block from the midpoint of that track and attempting to locate audio data that matches the Track Alignment Guide Data supplied by the server. If the track alignment data is not found, the client reads and searches adjoining blocks until the alignment data is found or a predefined number of blocks have been searched.

The client then computes the offset between the expected location of the track alignment data and the apparent location. After adjusting the location of the requested

audio data region by the alignment offset computed, the client reads the audio data from the disc and includes it in the Verification Request message. The client includes the TOC data in the Verification Request message since the server preserves no client state. The Encoded State Information included by the server in the Verification Response message is copied by the client unmodified into the Verification Request message. The Verification Request message is sent to the server. In an alternate embodiment, the client state information is maintained by the server for the duration of the client verification session and is not sent to or received from the client.

In block 317, the server receives the Verification Request message from the client and proceeds to extract the Key Region data. Verification Request Processing is then performed by the server upon receipt of a Verification Request message from the client. The Encoded State Information is extracted, decoded, and used to restore the state of the verification process. For each key region supplied by the client, the server tests the client-supplied data against the corresponding Key Data stored in the disc's entry in the Verification Table. Any data supplied by the client for a decoy key region is discarded.

The server then attempts to locate the actual key region within the enlarged key region data supplied by the client by locating the region that provides the greatest number of values that match the corresponding values in the Key Alignment Guide Data. The server computes a hash value, using the SHA-1 algorithm, of the key region identified in the alignment step. This hash value is compared with the value stored with the Key Data in the disc's entry in the Verification Table. If the values match exactly, the verification is successful, and the server constructs a Verified Response message. On the other hand, if the values do not match exactly, a Key Songprint is computed by the server.

In block 319, a Key Songprint is computed from the key region identified in the alignment step. An RMS difference is computed between the corresponding individual byte values of the songprint computed from the client-supplied data and the songprint that is stored with the Key Data in the disc's entry in the Verification Table. If the RMS difference is less than or equal to a predefined threshold value, the verification is successful and the process follows the Yes path from block 319 to block 321 where the server constructs a Verified Response message.

Returning to block 319, if the server determines the RMS difference exceeds the threshold, the process continues to block 323 and if one or more of the key candidates selected during Initial Request Processing have not yet been requested from the client, the process follows the Yes Path from block 323 to block 313 and the server proceeds to construct a new Verification Response message.

Returning to block 321, the Verified Response message is constructed by the server upon completion of a successful verification. The server includes identifying information for the verified disc including, for example, the disc's title and artist. Additional information is included as required by the overall application.

The server also computes the offset between the expected location of the key region within the enlarged key region and the actual location. This offset value is included in the Verified Response message to enable the client to adjust data read operations in future verifications. The server computes and encrypts authorization data, as required by the overall application, which the client can present to third-parties as credentials certifying that the disc has been verified. The Verified Response message is sent to the client.

Returning to block 323, if the RMS difference exceeds the threshold and all key candidates have been exhausted, the verification fails. The process then follows the No path to block 325 where a Not Verified Response message is constructed by the server

upon failing to locate in the Identification Table an entry that acceptably matches the disc being verified.

The client may also be programmed to respond in a particular manner to any of the system's messages, including a Verified message, a Not Verified message, or a Not Found message. For example, if the CD is verified, the client may be programmed to display information about the CD, or to automatically play the CD.